

CLAIMS

What is claimed is:

1. A correlating debugger, comprising:
  - a first logic configured to receive a first data set from a hardware analyzer that is configurable to analyze a hardware device;
  - a second logic configured to receive a second data set from a software analyzer that is configurable to analyze a software component;
  - a third logic configured to receive a binding data from the hardware analyzer or the software analyzer, where the binding data facilitates synchronizing the first data set and the second data set; and
  - a fourth logic operably connected to the first logic and the second logic, the fourth logic being configured to:
    - receive a signal that indicates that an interaction between the hardware device and the software component has occurred; and
    - upon receiving the signal, to selectively store elements of the first data set and the second data set in a time-ordered data set.
2. The correlating debugger of claim 1, where the first data set includes one or more of, timing information, state information, state change information, interrupt information, a register value, a memory location value, and voltage information.
3. The correlating debugger of claim 1, where the second data set includes one or more of, timing information, state information, state change information, interrupt information, a register value, a memory location value, a value for a data variable, information about a function call, information about a subroutine call, information about a method call, information about a stack, information about a heap, and information about a procedure call.
4. The correlating debugger of claim 1, where the hardware device comprises one of, a Universal Serial Bus (USB) device, a Small Computer Systems Interface (SCSI) device, an Industrial Standard Architecture (ISA) device, an Extended Industrial Standard Architecture (EISA) device, a Peripheral Component Interconnect (PCI) device, a PCI Express (PCIE)

device, a Serial Advanced Technology Attachment (SATA) device, an Infiniband device, an Ethernet device, a 1394 device, and a Microchannel Architecture (MSA) device.

5. The correlating debugger of claim 1, where the software component comprises one of, a device driver, and an application.

6. The correlating debugger of claim 1, where the binding data comprises a hardware identifier.

7. The correlating debugger of claim 1, where the signal comprises a non-maskable interrupt.

8. The correlating debugger of claim 1, where the time-ordered data set includes one or more first elements of the first data set and one or more second elements of the second data set, where the first elements are arranged together with the second elements in order based on time.

9. The correlating debugger of claim 1, where the software component is configured to run on a first processor and a processor executable instructions associated with the first logic, the second logic, the third logic, or the fourth logic are configured to run on a second processor.

10. The correlating debugger of claim 1, where the software component and processor executable instructions associated with one or more of, the first logic, the second logic, the third logic, and the fourth logic are configured to run on a first processor.

11. The correlating debugger of claim 1, where the software analyzer comprises a kernel debugger.

12. A system, comprising:

    a first logic configured to receive a first data set from a hardware analyzer that is configurable to analyze a USB hardware device, where the first data set includes one or more

of, timing information, state information, state change information, interrupt information, a register value, a memory location value, and voltage information;

a second logic configured to receive a second data set from a software analyzer that is configurable to analyze a device driver, where the second data set includes one or more of, timing information, state information, state change information, interrupt information, a register value, a memory location value, a value for a data variable, information about a function call, information about a subroutine call, information about a method call, information about a stack, information about a heap, and information about a procedure call;

a third logic configured to receive a hardware identification number from the software analyzer or the software component, where the hardware identification number facilitates synchronizing the first data set with the second data set; and

a fourth logic operably connected to the first logic and the second logic, the fourth logic being configured:

to receive a non-maskable interrupt from the hardware analyzer that indicates that an interaction between the USB hardware device and the device driver has occurred; and

upon receiving the non-maskable interrupt, to selectively store the first data set and elements of the second data set together, in a time-ordered data set.

13. A correlating debugger, comprising:

means for receiving data from a hardware analyzer that is configured to analyze a piece of computer hardware;

means for receiving data from a software analyzer that is configured to analyze a device driver configured to perform one or more of, processing interrupts from the piece of computer hardware, polling the piece of computer hardware, and polling a bus by which the piece of computer hardware and the software analyzer are operably connected;

means for coordinating the analysis of the hardware analyzer and the software analyzer so that data received from the hardware analyzer and data received from the software debugger can be merged in a time-ordered set of data; and

means for initiating writing data received from the hardware analyzer and data received from the software analyzer to the time-ordered set of data.

14. A method, comprising:

establishing a relationship in a debugger between a hardware device and a software component that will perform a software operation related to the hardware device;

configuring a software analyzer to collect a first data set related to the software component as the software component performs the software operation and to cause the first data set to be delivered to the debugger;

configuring a hardware analyzer to collect a second data set related to the hardware device as the hardware device performs a hardware operation and to cause the second data set to be delivered to the debugger;

detecting a first event that signals a beginning of the software operation and controlling the software analyzer to begin delivering the first data set to the debugger;

detecting a second event that signals a beginning of the hardware operation and controlling the hardware analyzer to begin delivering the second data set to the debugger; and

selectively storing together, in order, elements of the first data set and elements of the second data set, where the order is based, at least in part, on the time at which an event associated with generating an element occurred.

15. The method of claim 14, where establishing the relationship includes:

identifying a hardware number known to the software component;

communicating the hardware number to the hardware analyzer;

correlating the software component and a hardware device based on the hardware number; and

configuring the hardware analyzer to analyze the hardware device.

16. The method of claim 15, where the hardware number is based on an I/O request packet (IRP).

17. The method of claim 14, where configuring the software analyzer includes one or more of:

identifying one or more types of data available in the software component to be reported on by the software analyzer; and

establishing a frequency with which the one or more types of data will be sampled.

18. The method of claim 14, where configuring the hardware analyzer includes or more of, identifying one or more types of data available in the hardware device to be reported on by the hardware analyzer, and identifying one or more types of events from the hardware device to be reported on by the hardware analyzer.
19. The method of claim 14, where the software component comprises a device driver and the first event comprises an I/O request packet (IRP) arriving at a pre-determined, configurable level in the device driver.
20. The method of claim 14, where the second event comprises a state change on an operable connection between the hardware device and the software component.
21. The method of claim 14, where selectively storing together elements of the first data set and elements of the second data set includes selectively adding an element of the second data set to the first data set.
22. A computer-readable medium storing processor executable instructions operable to perform a method, the method comprising:
  - establishing a relationship in a debugger between a hardware device and a software component that will perform a software operation related to the hardware device by:
    - identifying a hardware number known to the software component;
    - communicating the hardware number to the hardware analyzer;
    - correlating the software component and a hardware device based on the hardware number; and
    - configuring the hardware analyzer to analyze the hardware device;
  - configuring a software analyzer to collect a first data set as the software component performs the software operation and to deliver the first data set to the debugger, where configuring the software analyzer includes one or more of, identifying one or more types of data available in the software component to be reported on by the software analyzer, and establishing a frequency with which the one or more types of data will be sampled;
  - configuring a hardware analyzer to collect a second data set related to the hardware device as the hardware device performs a hardware operation and to deliver the second data set to the debugger, where configuring the hardware analyzer includes or more of, identifying

one or more types of data available in the hardware device to be reported on by the hardware analyzer, and identifying one or more types of events from the hardware device to be reported on by the hardware analyzer;

detecting a first event that signals the beginning of the software operation and controlling the software analyzer to begin delivering the first data set to the debugger, where the software component comprises a device driver and the first event comprises an I/O request packet arriving at a pre-determined, configurable level in the device driver;

detecting a second event that signals the beginning of the hardware operation and controlling the hardware analyzer to begin delivering the second data set to the debugger, where the second event comprises a state change on an operable connection between the hardware device and the software component; and

selectively storing the first data set and the second data set together, in order, where the order is based, at least in part, on the time at which an event associated with generating a member of the first data set or a member of the second data set occurred.

23. A method, comprising:

engaging a hardware analyzer to analyze a piece of computer hardware;

engaging a software analyzer to analyze a piece of computer software that will service interrupts for the piece of computer hardware;

binding the hardware analyzer to the software analyzer to facilitate storing data received from the hardware analyzer and the software analyzer together, in order, based on time of occurrence;

receiving an event that indicates that data storage should begin; and

storing data received from the hardware analyzer and the software analyzer together, in order, based on time of occurrence.

24. The method of claim 23, where binding the hardware analyzer to the software analyzer includes programming the hardware analyzer to analyze a piece of computer hardware identified by a hardware number in an I/O request packet (IRP) available to the piece of software.

25. The method of claim 23, where binding the hardware analyzer to the software analyzer includes establishing a logic connection in a correlating debugger configured to receive data from the hardware analyzer and the software analyzer.
26. The method of claim 23, where binding the hardware analyzer to the software analyzer includes establishing a physical connection at a correlating debugger configured to receive data from the hardware analyzer and the software analyzer.
27. In a computer system having a graphical user interface comprising a display and a selection device, a method of providing and selecting from a set of data entries on the display, the method comprising:
  - retrieving a set of data entries, where a data entry represents an operation associated with ordering data received from a hardware analyzer configured to analyze a hardware device and a software debugger configured to analyze a device driver configured to perform one or more of, servicing interrupts from the hardware device, polling the hardware device and polling a bus by which the piece of computer hardware and the software debugger are operably connected;
  - displaying the set of data entries on the display;
  - receiving a data entry selection signal indicative of the selection device selecting a selected data entry; and
  - in response to the data entry selection signal, initiating an operation associated with the selected data entry.
28. A computer-readable medium having stored thereon a data structure comprising:
  - a first field containing data received from a hardware analyzer configured to analyze a hardware device;
  - a second field containing data received from a software debugger configured to analyze a software component that will perform one or more of, servicing interrupts related to the hardware device and polling the hardware device; and
  - a third field containing data derived from the first field and the second field, where the data in the third field is arranged in order, based on time.

29. A set of application programming interfaces embodied on a computer-readable medium for execution by a computer component in conjunction with ordering debug information, comprising:

- a first interface for communicating data associated with a hardware analyzer or a hardware device being analyzed by the hardware analyzer;
- a second interface for communicating data associated with a software debugger or a software component being analyzed by the software debugger; and
- a third interface for communicating data that facilitates binding the software debugger or the software component with the hardware analyzer or the hardware device to facilitate ordering debug information produced by the hardware analyzer and the software analyzer.

30. A system for correlating debug data, comprising:

- a first logic configured to receive a hardware related debug data from a hardware analyzer;
- a second logic configured to receive a software related debug data from a software analyzer;
- means for establishing a relationship between the hardware analyzer and the software analyzer to facilitate correlating data received by the first logic and data received by the second logic into a time-ordered set of data.